UMM Versioning Design

Goal

Provide a way to update the UMM schemas without breaking clients.

Traceability

⚠ CMR-2354 - JIRA project doesn't exist or you don't have permission to view it.

Rationale

The UMM schema will evolve over time. We need the ability to ingest and store JSON metadata according to one version of the schema, and then later load it according to the latest version. This problem is similar to database migration, except that we are migrating metadata on demand, instead of updating a database schema in-place with one-off operations. We must maintain previous versions of the UMM schema and make it available to consumers (like MMT) so that they can adequately respond to changes. This approach will allow CMR to evolve faster and give clients room to adapt to changes on their own terms.

Mitigating the Impact of Breaking Changes

Currently, changing portions of the UMM schema can lead to problems for CMR users (e.g. MMT). For example: changing an existing property from a simple object to an array property under a new name (e.g. *TilingIdentificationSystem* to TilingIdentificationSystem**s**, plural) would break client apps that were developed against the schema prior to the change.

To work around this issue, a user like MMT would always explicitly specify the version of UMM data that they are requesting or uploading via the *Accept* or *Content-Type* headers. This allows a client to continue dealing with the data they are used to, even if the UMM schema has been updated to a newer version.

Approach

JSON Schema Files

We will store JSON schema files in a directory named after the UMM version they represent. For example:

```
/resources/json-schemas/umm/1.1/common.json
/resources/json-schemas/umm/1.1/collection.json
/resources/json-schemas/umm/1.2/common.json
/resources/json-schemas/umm/1.2/collection.json
```

Individual JSON schemas will use relative references to refer to other schemas of a specific version.

```
"$ref": "../1.1/common.json#/definitions/ShortNameType"
```

A new route will serve an HTML page listing the JSON schema files for clients like MMT developers.

API Changes

For HTTP API clients interacting with the CMR, we will augment the UMM JSON media type with version numbers to explicitly specify the UMM version that is being sent or received. We will use media type parameters to explicitly specify a version, and assume the latest version if no version parameter is provided.

To conform to media type best practices, we will update the UMM media type to use the "vendor" tree, with a "vnd.nasa" prefix in the subtype to denote that UMM is a NASA media type.

Media Type	Type	Subtype	Suffix	Version Parameter
application/vnd.nasa.cmr.umm.collection+json; version=1.2	application	vnd.nasa.cmr.umm.collection	json	1.2

Specifying or omitting the version parameter in a request has different implications in different contexts. The following examples assume the current version of the UMM schema is **1.2:**

API Action	Request Accept Header	Request Content-Type Header	Response Content-Type Header	Result

Ingesting JSON metadata	n/a	application/vnd.nasa.cmr.umm+json; version=1.2	application/vnd.nasa.cmr.umm+json; version=1.2	The metadata is parsed and validated according to the specified version **1.2** before being converted to in-memory UMM records for ingest and indexing; the specified version is stored in Metadata DB and the JSON metadata is returned exactly as it was ingested.
Ingesting JSON metadata	n/a	application/vnd.nasa.cmr.umm+json	application/vnd.nasa.cmr.umm+json; version=1.2	The latest version of UMM is assumed, and metadata is parsed and ingested accordingly. **Question:** should we support this? Is MMT the only client using the API to ingest UMM JSON?
Requesting a concept ingested as UMM JSON	application/vnd.nasa.cmr.umm+json	n/a	application/vnd.nasa.cmr.umm+json; version=1.2	The concept metadata is returned as-is, and the Content-Type header contains a media type with a version parameter corresponding to the version used when the concept was initially ingested.
Requesting a concept ingested as UMM JSON	application/vnd.nasa.cmr.umm+json; version=1.2	n/a	application/vnd.nasa.cmr.umm+json; version=1.2	The concept's JSON metadata is parsed and converted to the desired version using the above mentioned migration code before being encoded as JSON, and returned with a Content-Type header containing a version parameter indicating the requested version.
Requesting a concept initially ingested as XML as UMM JSON	application/vnd.nasa.cmr.umm+json	n/a	application/vnd.nasa.cmr.umm+json; version=1.2	The latest version of UMM is assumed and the XML metadata is parsed as UMM records and encoded as JSON (as currently implemented)
Requesting a concept initially ingested as XML as UMM JSON	application/vnd.nasa.cmr.umm+json; version=1.1	n/a	application/vnd.nasa.cmr.umm+json; version=1.1	The XML metadata is parsed into UMM records according to the latest schema and migrated to the requested UMM version before being encoded as JSON.

Error rendering macro 'pageapproval' : null